

Ruby Programming

Duration: 4 Days

Description

This is a technical course that introduces the Ruby programming language. It is an instructor led presentation and hands on exercises course using MS Windows, but is equally applicable to other platforms such as Linux. The course material covers versions of Ruby from 1.9 onwards, but significant changes from earlier versions will be mentioned.

Delegates will learn how to

- Use the Ruby interactive interpreter to write and run Ruby programs
- Understand Ruby language elements
- Exploit the rich library of Ruby libraries and modules
- Recognize simple and complex variable types and select appropriately
- Use Ruby operators and built-in methods
- Understand procedural control flow in Ruby
- Program file input/output, including persistent data objects.
- Create well organized and efficient code using classes, methods and modules
- Use Object Oriented programming techniques in Ruby.
- Build robust applications with error trapping and reporting
- Run and control other programs from Ruby
- Take advantage of multi-core processors with multiprocessing.

Outline

Introduction to Ruby

- What is Ruby?
- Background
- Ruby versions and implementations
- Installing Ruby
- Running Ruby interactively
- Ruby programs
- A few definitions

- Ruby documentation and help
- Anatomy of a Ruby program
- Ruby style
- Comments
- Delimiters
- Simple terminal IO
- Libraries and Classes

Fundamental Variables

- Ruby is Object Oriented
- A class is not a type!
- Ruby assignments
- Avoiding alias issues
- Assignment shortcuts
- Ruby variables
- Variable names
- Ruby literal types
- Ruby numerics
- Ruby Arrays introduced
- Parallel assignments
- Ranges
- Ruby Hashes introduced
- Constants
- Symbols
- Type conversion methods
- A note on Exception Handling
- Predefined variables

Basic Flow Control

- Ruby comparison operators
- Logical operators
- What is truth?
- Ruby conditionals - if
- Variations on if
- Ruby conditionals - else and elsif
- Ternary and defined?
- Multi-way selection - case/when
- at_exit methods
- Unconditional flow control

- Blocks
- A conditional is a block

String Handling

- Ruby strings
- Encoding
- String concatenation
- Slicing strings
- String methods
- String tests
- Interpolation
- String formatting
- sprintf
- printf
- % operator
- String formatting methods
- String split and Array join

Collections

- Ruby types - reminder
- Ruby Arrays
- Slicing Arrays
- Stack and deque type operations
- Adding items to an Array
- Removing items from an Array
- Sorting
- Array set operations
- Array methods
- Ruby Hashes
- Hash keys
- Testing and merging hashes
- Sets

Loops and iterators

- Code blocks
- Passing parameters to blocks
- Using blocks
- While loops
- Variations on while

- Loop control statements
- The basic loop
- For loops
- Counting loops
- Enumerable
- Map and grep

Regular Expressions

- Ruby regular expressions
- Elementary extended RE meta-characters
- Regular expression objects
- Pattern match syntax
- Regular expression substitution
- Matching alternatives
- Anchors
- Class shortcuts
- Options
- Quantifiers
- Minimal matches
- Back-references
- Side-effect variables
- Sub-patterns and MatchData
- Global matches - scan
- Using gsub with a Hash

Data Storage and File Handling

- Standard streams
- IO objects and methods
- Reading files
- Filter Programs - ARGF
- Command-line ARGF processing
- Binary mode
- Writing to files
- Random access